STIC Search Report
EIC 2100

STIC Database Tracking Number: 131885

TO: Tuan A. Vu **Location: 5Y18** **Art Unit : 2124** **Wednesday, September 08, 2004** **Case Serial Number: 09/654115**	**From: David Holloway** **Location: EIC 2100** **PK2-4B30** **Phone: 308-7794** **david.holloway@uspto.gov**

Search Notes

Dear Examiner Vu,

Attached please find your search results for above-referenced case.
Please contact me if you have any questions or would like a re-focused search.

David

STIC EIC 2100 Search Request Form

131885

Today's Date:

What date would you like to use to limit the search?

Priority Date: Other:

Name TUAN VU

AU 2124 Examiner # 79545

Room # 5Y18 Phone 305 7207

Serial # 09/654,115

Format for Search Results (Circle One):

PAPER DISK EMAIL

Where have you searched so far?

USP DWPI EPO JPO ACM IBM TDB

IEEE INSPEC SPI Other _____

Is this a "Fast & Focused" Search Request? (Circle One) YES NO

A "Fast & Focused" Search is completed in 2-3 hours (maximum). The search must be on a very specific topic and meet certain criteria. The criteria are posted in EIC2100 and on the EIC2100 NPL Web Page at http://ptoweb/patents/stic/stic-tc2100.htm.

What is the topic, novelty, motivation, utility, or other specific details defining the desired focus of this search? Please include the concepts, synonyms, keywords, acronyms, definitions, strategies, and anything else that helps to describe the topic. Please attach a copy of the abstract, background, brief summary, pertinent claims and any citations of relevant art you have found.

System for selectively enabling only certain program/instructions/data (e.g. intermediate code) to be cached, thus reducing cache hits/thrashing, such system including a compiler for adding bits to each instruction end to act as markers. those markers will enable the runtime to determine whether an instruction is to be cached, more specifically:

(1) whether to cache

(2) which cache (level 1 or level 2 cache?)

(3) is it a write-back or a write-through.

STIC Searcher David Hamilton Phone 308-7794

Date picked up 9-8-04 Date Completed 9-8-04

Search and Information Resources Administration

```
Set     Items   Description
S1     287368   CACH? OR BUFFER? OR TEMPORAR?(N)(STOR? OR SAVE?)
S2     125680   (SPECIFIC? OR PARTIAL OR PART OR CERTAIN? OR DESIGNAT? OR -
             SEGMENT?)(2N)(CODE OR PROGRAM? OR INSTRUCTION? OR DATA OR SUB-
             PROGRAM?)
S3     726810   MARKER? OR FLAG OR FLAGS OR TAG OR TAGS OR INDICAT? OR LAB-
             EL?
S4    1326186   APPEND? OR ADD()ON OR BITS OR BIT OR INSERT?
S5      31704   (LEVEL? OR WHICH? OR TIER? OR AREA? OR WHERE? OR ADDRESS?)-
             (2N)S1
S6        996   (COPY OR WRITE)()(BACK OR THROUGH) OR WRITETHROUGH OR WRIT-
             EBACK OR COPYBACK? OR COPYTHROUGH?
S7      10390   S1 AND S2
S8         75   S7 AND S3(2N)S4
S9       1565   S7 AND S5
S10        35   S7 AND S6
S11        15   S8 AND S5
S12        19   S1(5N)S2(5N)S3(2N)S4
S13        17   S1(5N)S2 AND S6
S14        48   S13 OR S11 OR S12
S15        44   S14 AND IC=G06F?
S16        44   IDPAT (sorted in duplicate/non-duplicate order)
S17        44   IDPAT (primary/non-duplicate records only)
S18     18159   S1(2N)(CONTROL? OR MANAGE? OR ADMINIST? OR MONITOR?)
S19       206   S2(5N)S18
S20        77   S19 AND (S3 OR S4 OR S5 OR S6)
S21         6   S20 AND IC=G06F-009?
S22         5   S21 NOT S17
S23         5   IDPAT (sorted in duplicate/non-duplicate order)
S24         5   IDPAT (primary/non-duplicate records only)
File 347:JAPIO Nov 1976-2004/May(Updated 040903)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &UP=200457
         (c) 2004  Thomson Derwent
```

24/5/1 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013224815 **Image available**
WPI Acc No: 2000-396689/200034
XRPX Acc No: N00-434928

Pipelined data processing system monitors identifier provided by functional unit and stores monitored result in buffer

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: LEE H G; CHEONG H; LE H Q
Number of Countries: 002 Number of Patents: 003
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
KR 99013425	A	19990225	KR 9822438	A	19980615	200034	B
US 6070235	A	20000530	US 97892589	A	19970714	200056	
KR 305935	B	20011019	KR 9822438	A	19980615	200234	

Priority Applications (No Type Date): US 97892589 A 19970714
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
KR 99013425	A			G06F-009/00	
US 6070235	A		19	G06F-015/00	
KR 305935	B			G06F-009/00	Previous Publ. patent KR 99013425

Abstract (Basic): US 6070235 A

NOVELTY - A history buffer is coupled to logic unit for receiving a target identifier of **specific instruction** . The **buffer monitors** identifier provided by functional unit, and stores the monitored result, when control signal output from functional unit is in first logical state. The control signal is switched to second logical state, to **indicate** that the monitored result is unavailable for storage within a register.

DETAILED DESCRIPTION - The target identifier identifies a register to be accessed by the instruction. An INDEPENDENT CLAIM is also included for method for capturing data in data processing system.

USE - For data processing.

ADVANTAGE - Enables ensuring that data is captured in correct chronological order and avoids corruption of historical values of registers.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of pipeline processor.

pp; 19 DwgNo 2/4

Title Terms: PIPE; DATA; PROCESS; SYSTEM; MONITOR; IDENTIFY; FUNCTION; UNIT ; STORAGE; MONITOR; RESULT; BUFFER
Derwent Class: T01
International Patent Class (Main): **G06F-009/00** ; G06F-015/00
File Segment: EPI

24/5/3 **(Item 3 from file: 350)**
DIALOG(R)File 350:Derwent WPIX

011113770 **Image available**
WPI Acc No: 1997-091695/199709
XRPX Acc No: N97-075602
Computer system with cache system - has cache controller which **directs** cache **memory to** designate data **which are returned by CPU to main memory to another address, in response to warning signal received from CPU**
Patent Assignee: TOSHIBA KK (TOKE)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
JP 8328953	A	19961213	JP 95130457	A	19950529	199709	B

Priority Applications (No Type Date): JP 95130457 A 19950529
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
JP 8328953	A		10	G06F-012/08	

Abstract (Basic): JP 8328953 A
The computer system has a CPU (1) which outputs a warning signal to **indicate** first memory data access enforcement or indirect addressing instruction. The CPU outputs an access control command to a main memory (4). A cache memory (3) is accessed by the CPU only when it holds the address of the data to be accessed in the main memory.
The data are returned to the main memory when the accessed data are altered. A cache controller (2) controls the cache memory so that the returned data will be stored in a certain address.
ADVANTAGE - Shortens time of memory access through cache memory; enables efficient memory access.
Dwg.1/1
Title Terms: COMPUTER; SYSTEM; CACHE; SYSTEM; CACHE; CONTROL; DIRECT; CACHE ; MEMORY; DESIGNATED; DATA; RETURN; CPU; MAIN; MEMORY; ADDRESS; RESPOND; WARNING; SIGNAL; RECEIVE; CPU
Derwent Class: T01
International Patent Class (Main): G06F-012/08
International Patent Class (Additional): **G06F-009/38**
File Segment: EPI

```
Set     Items   Description
S1     287368   CACH? OR BUFFER? OR TEMPORAR?(N)(STOR? OR SAVE?)
S2     125680   (SPECIFIC? OR PARTIAL OR PART OR CERTAIN? OR DESIGNAT? OR -
             SEGMENT?)(2N)(CODE OR PROGRAM? OR INSTRUCTION? OR DATA OR SUB-
             PROGRAM?)
S3     726810   MARKER? OR FLAG OR FLAGS OR TAG OR TAGS OR INDICAT? OR LAB-
             EL?
S4    1326186   APPEND? OR ADD()ON OR BITS OR BIT OR INSERT?
S5      31704   (LEVEL? OR WHICH? OR TIER? OR AREA? OR WHERE? OR ADDRESS?)-
             (2N)S1
S6        996   (COPY OR WRITE)()(BACK OR THROUGH) OR WRITETHROUGH OR WRIT-
             EBACK OR COPYBACK? OR COPYTHROUGH?
S7      10390   S1 AND S2
S8         75   S7 AND S3(2N)S4
S9       1565   S7 AND S5
S10        35   S7 AND S6
S11        15   S8 AND S5
S12        19   S1(5N)S2(5N)S3(2N)S4
S13        17   S1(5N)S2 AND S6
S14        48   S13 OR S11 OR S12
S15        44   S14 AND IC=G06F?
S16        44   IDPAT (sorted in duplicate/non-duplicate order)
S17        44   IDPAT (primary/non-duplicate records only)
File 347:JAPIO Nov 1976-2004/May(Updated 040903)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &UP=200457
         (c) 2004  Thomson Derwent
```

17/5/5 **(Item 5 from file: 350)**
DIALOG(R)File 350:Derwent WPIX

012203171 **Image available**
WPI Acc No: 1999-009277/199901
Related WPI Acc No: 1998-346695
XRPX Acc No: N99-006748

Cache apparatus for magnetic disk sub-system - controls access of disk units corresponding to preset RAID operating mode, when disk array is accessed by read and write cache control units

Patent Assignee: FUJITSU LTD (FUIT)
Inventor: IWATANI S; YORIMITSU K
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5835940	A	19981110	US 94266737	A	19940627	199901	B
			US 96606081	A	19960223		

Priority Applications (No Type Date): JP 93256216 A 19931014
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5835940	A		62	G06F-012/08	Div ex application US 94266737

Abstract (Basic): US 5835940 A

The apparatus has a disk array with several disk units into which data read-write is performed. A **cache** memory stores **part** of **data** in the disk array. A **cache** management unit manages storing state of cache memory based on a hash table indicating contents of cache registration. An LRU table includes cache blocks having effective data that are sequentially linked, based on a registration order. The cache block which is used most recently is set as a leading head block. A read cache control unit reads data from the cache memory to a higher order apparatus, when a read request is received from the higher order apparatus. When relevant data is not found in the cache memory for transfer, data is read from the disk array to the higher order apparatus.

A write cache control unit registers a management information when write request is received from the higher order apparatus. Data is written into the cache memory corresponding to the management information. A **write back** control unit extracts data which is not yet stored into the disk array from cache memory. The extracted data is written back, when predetermined **write back** conditions are satisfied. A disk array control unit controls access to several disk units corresponding to a preset RAID operating mode, when disk array is accessed by the read and write cache control unit.

ADVANTAGE - Improves accessing speed. Prevents fault occurrence, thereby preventing loss of data.

Dwg.8/46

Title Terms: CACHE; APPARATUS; MAGNETIC; DISC; SUB; SYSTEM; CONTROL; ACCESS ; DISC; UNIT; CORRESPOND; PRESET; RAID; OPERATE; MODE; DISC; ARRAY; ACCESS; READ; WRITING; CACHE; CONTROL; UNIT
Derwent Class: T01
International Patent Class (Main): **G06F-012/08**
International Patent Class (Additional): **G06F-013/00**
File Segment: EPI

17/5/7 **(Item 7 from file: 350)**
DIALOG(R)File 350:Derwent WPIX

011996715 **Image available**
WPI Acc No: 1998-413625/199835
Related WPI Acc No: 1999-009272
XRPX Acc No: N98-321991

Computer memory system with cache memory e.g. for hard disk drive, DRAM, SRAM - in which specific data requested by processor is provided to processor from cache memory array when cache line is not completely filled

Patent Assignee: INTEGRATED DEVICE TECHNOLOGY (INTE-N)
Inventor: GASKINS D; HENRY G

Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5781926	A	19980714	US 96650733	A	19960520	199835	B

Priority Applications (No Type Date): US 96650733 A 19960520
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5781926	A		15	G06F-012/12	

Abstract (Basic): US 5781926 A

The system includes a main memory for storing data used by a processor (502). A cache memory (504) connected to the main memory and processor also stores data used by the processor. The cache memory comprises a cache memory array which has multiple cache lines (528). Each of the cache lines has multiple sub-cache line locations (530,532,534,536) for storing data. A cache directory (506) stores tag information associated with the data stored in the cache memory array. A cache enable unit provides multiple sub-cache line enable bits associated with the sub-cache line location in any one of the cache line, for allowing sub- cache lines location to be directly written into main memory without requiring intermediate storage in a cache buffer.

The cache enable unit has a cache logic (508) for reading **tag** information and the sub- **cache** line enable **bit** to determine whether **specific data** requested by processor is stored in the **cache** memory array. The **cache** enable logic is also used for providing the **specific data** to the processor when the **cache** line is not completely filled.

ADVANTAGE - Avoids use of pipeline stall for filling instruction queue, thereby improving performance. Reduces processing time by allowing access to sub-cache lines before filling entire cache line. Updates several sub cache line enable bits within short time.

Dwg.5/7

Title Terms: COMPUTER; MEMORY; SYSTEM; CACHE; MEMORY; HARD; DISC; DRIVE; DRAM; SRAM; SPECIFIC; DATA; REQUEST; PROCESSOR; PROCESSOR; CACHE; MEMORY; ARRAY; CACHE; LINE; COMPLETE; FILLED
Derwent Class: T01
International Patent Class (Main): **G06F-012/12**
File Segment: EPI

17/5/11 **(Item 11 from file: 350)**
DIALOG(R)File 350:Derwent WPIX

010792147 **Image available**
WPI Acc No: 1996-289100/199630
Cache control unit - includes boundary identification logic to determine nature of byte and anticipation buffer loaded with sequentially anticipated prefetched instructions
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: CIAVAGLIA S J; CONOR S M; KARTSCHOKE P D; MAHIN S W; MOULTON L H; RICH S E
Number of Countries: 004 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 718758	A2	19960626	EP 95480163	A	19951121	199630	B
EP 718758	A3	19961002	EP 95480163	A	19951121	199645	
US 5625787	A	19970429	US 94360520	A	19941221	199723	
			US 95440034	A	19950512		
US 5640526	A	19970617	US 94360520	A	19941221	199730	
US 5644744	A	19970701	US 94360520	A	19941221	199732	
			US 95440035	A	19950512		

Priority Applications (No Type Date): US 94360520 A 19941221; US 95440034 A 19950512; US 95440035 A 19950512
Cited Patents: 1.Jnl.Ref; EP 159712; EP 498654; GB 2263987; WO 8909442
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 718758	A2	E	11	G06F-009/38	
Designated States (Regional): DE FR GB					
US 5625787	A		11	G06F-009/38	Div ex application US 94360520
US 5640526	A		10	G06F-009/312	
US 5644744	A		10	G06F-009/312	Div ex application US 94360520
EP 718758	A3			G06F-009/38	

Abstract (Basic): EP 718758 A
The cache control unit has a content addressable memory for storing most recent addresses of cache lines which have been accessed for instruction fetching for execution, the instructions being of variable length. Boundary identification logic is responsive to the content addressable memory for identifying instruction boundaries for each of the number of instructions held in cache.
An anticipation buffer holds a next instruction, the next instruction being located and identified by the content addressable memory and the boundary identification logic.
ADVANTAGE - Effective mechanism of minimised complexity to permit high speed out-of-order instruction execution in microprocessor architectures that permit store-to-the-instruction stream operations.
Dwg.1/5
Title Terms: CACHE; CONTROL; UNIT; BOUNDARY; IDENTIFY; LOGIC; DETERMINE; NATURE; BYTE; ANTICIPATE; BUFFER; LOAD; SEQUENCE; ANTICIPATE; INSTRUCTION
Derwent Class: T01
International Patent Class (Main): **G06F-009/312** ; **G06F-009/38**
File Segment: EPI

17/5/21 **(Item 21 from file: 347)**
DIALOG(R)File 347:JAPIO

06462368 **Image available**

DEVICE AND METHOD FOR CONTROLLING CACHE MEMORY

PUB. NO.: 2000-047942 [JP 2000047942 A]
PUBLISHED: February 18, 2000 (20000218)
INVENTOR(s): INOUE TAKEHIRO
APPLICANT(s): NEC CORP
APPL. NO.: 10-213172 [JP 98213172]
FILED: July 28, 1998 (19980728)
INTL CLASS: **G06F-012/12**

ABSTRACT

PROBLEM TO BE SOLVED: To prevent an overhead while considering the reference frequency of a data block at the time of replace.

SOLUTION: The cache memory controller of a set associative system is provided with a cache memory of plural ways installed in a processor so as to be used for exchanging data between a register and a main storage device and a control means for controlling this cache memory. In this case, a **cache tag** 10 to be stored in the **cache** memory is composed of a **data part** 11 for storing data, an address **tag** part 13 for storing the address of data, an LRU **bit** part 15 for storing information for grasping whether these data are just recently referred to or not and access frequency bit part 17 for storing information for grasping the reference frequency of referring to the data and the information of the said access frequency bit part is considered for a replace algorithm.

17/5/23 **(Item 23 from file: 347)**
DIALOG(R)File 347:JAPIO

04676695 **Image available**
CACHE DEVICE

PUB. NO.: 06-348595 [JP 6348595 A]
PUBLISHED: December 22, 1994 (19941222)
INVENTOR(s): KUSHIMA ICHIRO
UMINAGA MASAHIRO
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 05-163148 [JP 93163148]
FILED: June 07, 1993 (19930607)
INTL CLASS: [5] **G06F-012/08**
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units)

ABSTRACT

PURPOSE: To provide a **cache** device **which** can leave **specific data** preferentially in a **cache** .

CONSTITUTION: Each entry of the **cache** is provided with a reuse bit for preferentially inhibiting data from being expelled and when only one reuse bit (404) of two sets of entries is true as shown in the figure, the other entry is selected as an expelled entry. When the reuse bits of both the sets are true or false, an entry whose recent bit 402 is false is selected. The instruction word of a load instruction is provided with a **bit indicating** whether or not there is REUSE designation and when this bit is true, the reuse bit (404) of a **cache** entry including data accessed by the load instruction becomes true.

03086237 **Image available**
STORAGE DEVICE

PUB. NO.: 02-061737 [JP 2061737 A]
PUBLISHED: March 01, 1990 (19900301)
INVENTOR(s): FUKUDA HIROSHI
SAIE YASUHIKO
KIKUCHI TAKASHI
HATANO SUSUMU
OOISHI KANJI
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
HITACHI VLSI ENG CORP [489108] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 63-212626 [JP 88212626]
FILED: August 29, 1988 (19880829)
INTL CLASS: [5] **G06F-012/08**
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units)
JAPIO KEYWORD:R131 (INFORMATION PROCESSING -- Microcomputers & Microprocessers)
JOURNAL: Section: P, Section No. 1051, Vol. 14, No. 240, Pg. 60, May 22, 1990 (19900522)

ABSTRACT

PURPOSE: To shorten the waiting time of an MPU to improve the throughput of a system by providing a **flag bit** group **indicating** whether internal data is effective or not correspondingly to a block **buffer** .

CONSTITUTION: A block **buffer** 18 **where** one-block components of data can be stored at a time is provided between an internal data bus 17b of a storage device, for example, a **cache** memory 1 and a **data** memory **part** 12, and the unit of data transfer between the **cache** memory and the **data** memory **part** 12 is larger than that between the **cache** memory 1 and a main memory. A **flag bit** group in a register 19 is provided which corresponds to respective data stored in the block **buffer** 18 and indicates whether data is effective/ ineffective, and a decoder 19a is provided which can simultaneously select plural **flag bits** in accordance with the transfer data width at the time of terminating data transfer from the block **buffer** 18 to the data memory 12, and **flag bits** which are set one by one are simultaneously cleared.

17/5/40 **(Item 40 from file: 347)**
DIALOG(R)File 347:JAPIO

01877656 **Image available**
CONTROL SYSTEM FOR **BUFFER** MEMORY IN INPUT AND OUTPUT CONTROLLER

PUB. NO.: 61-091756 [JP 61091756 A]
PUBLISHED: May 09, 1986 (19860509)
INVENTOR(s): NAKAMURA HOSAKU
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 59-213888 [JP 84213888]
FILED: October 12, 1984 (19841012)
INTL CLASS: [4] **G06F-013/38**
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units)
JOURNAL: Section: P, Section No. 496, Vol. 10, No. 266, Pg. 130, September 11, 1986 (19860911)

ABSTRACT

PURPOSE: To **buffer** storage length data of optional length by stopping data reading operation from a **buffer** memory when a final data **flag bit** is detected.

CONSTITUTION: When data is written on a magnetic tape device (MTU)5 from a channel device (CH)3, an input/output controller 4 receives a write command and sets the next address of a final data **address** in a **buffer** memory **address** register 29. Then, CH3 sends data, which is stored in the **buffer** memory; 23. When final data of one record length is sent, '1' is sent as a final data indica tion signal together with the data. Consequently, the final data flag in the **buffer** memory is set to '1'. When the **buffer** memory 23 becomes full, reading opera tion is started and one transfer data is read out of the **buffer** memory 23. At this time, only the **data part** in a B data register 24 is sent to MTU5. Then, when the final data flag is '1', transfer to MTU5 is finished.

```
Set     Items   Description
S1     287368   CACH? OR BUFFER? OR TEMPORAR?(N)(STOR? OR SAVE?)
S2     125680   (SPECIFIC? OR PARTIAL OR PART OR CERTAIN? OR DESIGNAT? OR -
             SEGMENT?)(2N)(CODE OR PROGRAM? OR INSTRUCTION? OR DATA OR SUB-
             PROGRAM?)
S3     726810   MARKER? OR FLAG OR FLAGS OR TAG OR TAGS OR INDICAT? OR LAB-
             EL?
S4       6324   (APPEND? OR ADD()ON OR INSERT?)(2N)(BIT OR BITS OR S3)
S5        996   (COPY OR WRITE)()(BACK OR THROUGH) OR WRITETHROUGH OR WRIT-
             EBACK OR COPYBACK? OR COPYTHROUGH?
S6       9091   COMPILE?
S7          0   S1 AND S2 AND S3 AND S4 AND S5
S8         19   S1 AND S2 AND S4
S9          2   S1 AND S5 AND S6
S10        14   S1(4N)S2 AND S5
S11        35   S8 OR S9 OR S10
S12        20   S11 AND IC=G06F?
S13        20   IDPAT (sorted in duplicate/non-duplicate order)
S14        20   IDPAT (primary/non-duplicate records only)
File 347:JAPIO Nov 1976-2004/May(Updated 040903)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &U
```

14/5/4 (Item 4 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013473575
WPI Acc No: 2000-645518/200062
XRPX Acc No: N00-478345
Method of operating a multi-tasking computer system where each task is assigned its own cache **space.**
Patent Assignee: ANONYMOUS (ANON)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
RD 435104	A	20000710	RD 2000435104	A	20000620	200062	B

Priority Applications (No Type Date): RD 2000435104 A 20000620
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
RD 435104	A		2	G06F-000/00	

Abstract (Basic): RD 435104 A
NOVELTY - In a multi-tasking data processing system each task is assigned its own **cache** store. Each task can start without flushing any other task's **cache** and during task switching no **write - back** is required. Each task has its own identification number set by the operating system that selects the appropriate **cache** for the current task. All non-selected areas of **cache** can be left inactive to save power consumption.
USE - In multi-tasking data processing systems.
ADVANTAGE - Reduces **cache** charge/discharge overhead and makes possible **cache** swapping, **compiler** directed pre-charging and dynamic **cache** sizing. The cost of **cache** stores can be reduced as can power consumption.
pp; 2 DwgNo 0/1
Title Terms: METHOD; OPERATE; MULTI; COMPUTER; SYSTEM; TASK; ASSIGN; **CACHE** ; SPACE
Derwent Class: T01
International Patent Class (Main): **G06F-000/00**

14/5/7 (Item 7 from file: 350)
DIALOG(R)File 350:Derwent WPIX

010792147 **Image available**
WPI Acc No: 1996-289100/199630
Cache control unit - includes boundary identification logic to determine nature of byte and anticipation buffer loaded with sequentially anticipated prefetched instructions
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: CIAVAGLIA S J; CONOR S M; KARTSCHOKE P D; MAHIN S W; MOULTON L H; RICH S E
Number of Countries: 004 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 718758	A2	19960626	EP 95480163	A	19951121	199630	B
EP 718758	A3	19961002	EP 95480163	A	19951121	199645	
US 5625787	A	19970429	US 94360520	A	19941221	199723	
			US 95440034	A	19950512		
US 5640526	A	19970617	US 94360520	A	19941221	199730	
US 5644744	A	19970701	US 94360520	A	19941221	199732	
			US 95440035	A	19950512		

Priority Applications (No Type Date): US 94360520 A 19941221; US 95440034 A 19950512; US 95440035 A 19950512
Cited Patents: 1.Jnl.Ref; EP 159712; EP 498654; GB 2263987; WO 8909442
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 718758	A2	E	11	G06F-009/38	
Designated States (Regional): DE FR GB					
US 5625787	A		11	G06F-009/38	Div ex application US 94360520
US 5640526	A		10	G06F-009/312	
US 5644744	A		10	G06F-009/312	Div ex application US 94360520
EP 718758	A3			G06F-009/38	

Abstract (Basic): EP 718758 A

The cache control unit has a content addressable memory for storing most recent addresses of cache lines which have been accessed for instruction fetching for execution, the instructions being of variable length. Boundary identification logic is responsive to the content addressable memory for identifying instruction boundaries for each of the number of instructions held in cache.

An anticipation buffer holds a next instruction, the next instruction being located and identified by the content addressable memory and the boundary identification logic.

ADVANTAGE - Effective mechanism of minimised complexity to permit high speed out-of-order instruction execution in microprocessor architectures that permit store-to-the-instruction stream operations.

Dwg.1/5

Title Terms: CACHE; CONTROL; UNIT; BOUNDARY; IDENTIFY; LOGIC; DETERMINE; NATURE; BYTE; ANTICIPATE; BUFFER; LOAD; SEQUENCE; ANTICIPATE; INSTRUCTION
Derwent Class: T01
International Patent Class (Main): **G06F-009/312** ; **G06F-009/38**
File Segment: EPI

14/5/9 (Item 9 from file: 350)
DIALOG(R)File 350:Derwent WPIX

007828356 **Image available**
WPI Acc No: 1989-093468/198912
XRPX Acc No: N89-071158

Hierarchial memory system with reduced loss vulnerability - has separate criteria for replacement and write back **without replacement in accordance with age since first write algorithm**

Patent Assignee: UNISYS CORP (BURS)
Inventor: HAMSTRA J R
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 4811203	A	19890307	US 82354558	A	19820303	198912	B

Priority Applications (No Type Date): US 82354558 A 19820303
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 4811203	A		35		

Abstract (Basic): US 4811203 A

A linked list of are stored identifying of data in the cache memory which have been written-to in response to at least one write command from the host processor. One entry in the linked list corresponds to each segment in the cache memory that has been written to, the control unit is responsive to write commands from the host processor for making an entry in the linked list the first time a **segment** of **data** in the **cache** memory is written to, the entries being linked in the order in which their corresponding segments are first written to. A trickle device passes **segments** of **data** from the **cache** memory to the bulk memory.

The trickle device is responsive to the in the linked list for trickling segments of data to the bulk memory in the same order as their corresponding entries were made in the linked list. Replacement of the segments in cache memory is accomplished in order from the least recently used to the most recently used and trickling of segments is accomplished in the order of age since first being written to, the oldest written-to segment being trickled first.

USE - For reducing the vulnerability of a system to loss of data resulting from failure of a cache memory

Title Terms: MEMORY; SYSTEM; REDUCE; LOSS; VULNERABLE; SEPARATE; CRITERIA; REPLACE; WRITING; BACK; REPLACE; ACCORD; AGE; FIRST; WRITING; ALGORITHM
Derwent Class: T01
International Patent Class (Additional): **G06F-012/08** ; **G06F-013/12**
File Segment: EPI

14/5/10 (Item 10 from file: 347)
DIALOG(R)File 347:JAPIO

07522607 **Image available**
IMAGE GENERATING DEVICE

PUB. NO.: 2003-016438 [JP 2003016438 A]
PUBLISHED: January 17, 2003 (20030117)
INVENTOR(s): KANI MAMORU
APPLICANT(s): KYOCERA CORP
APPL. NO.: 2001-196942 [JP 2001196942]
FILED: June 28, 2001 (20010628)
INTL CLASS: G06T-001/60; **G06F-012/08**

ABSTRACT

PROBLEM TO BE SOLVED: To make the image forming speed higher while suppressing the rise in the cost of a product having an image generating device.

SOLUTION: When a cache access from a drawing means 10 to a cache system 20 has been generated, an image data control part 22-1 makes a hit/miss judgement on **specific data** in a **cache** array 21. When a miss is judged, the value of a **write - back** management bit regarding a requested cache class is confirmed. When the value of the **write - back** management bit is '1', a load request to a frame buffer 30 is issued. When the value of the **write - back** management bit is '0', on the other hand, a cache line is automatically generated by a line inside generation part 22-2 according to a background pattern.

14/5/15 (Item 15 from file: 347)
DIALOG(R)File 347:JAPIO

04332770 **Image available**
MULTIPROCESSOR SYSTEM, METHOD AND DEVICE FOR CONTROLLING CACHE MEMORY

PUB. NO.: 05-324470 [JP 5324470 A]
PUBLISHED: December 07, 1993 (19931207)
INVENTOR(s): YAMAGUCHI SHINICHIRO
KANEKAWA NOBUYASU
FUKUMARU HIROAKI
TANJI MASAYUKI
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 04-135145 [JP 92135145]
FILED: May 27, 1992 (19920527)
INTL CLASS: [5] **G06F-012/08** ; **G06F-012/08**
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)
JAPIO KEYWORD: R131 (INFORMATION PROCESSING -- Microcomputers & Microprocessers)
JOURNAL: Section: P, Section No. 1709, Vol. 18, No. 153, Pg. 89, March 14, 1994 (19940314)

ABSTRACT

PURPOSE: To keep the consistency of contents in each cache memory with a little hardware in the case of cache memory control for closely coupled multiprocessors.

CONSTITUTION: Each processor is provided with an **instruction** execution **part** 201, **copy back** type **cache** memory 202 and bus interface part 203 and when it is standby state since there is not bus use right at the time of writing data held in the cache memory 202 back to a main memory, a system bus 4 is monitored by a monitor circuit 204. When it is detected that data on the main memory in the same line as the **write back** object are reloaded by the other processor, the bus interface part 203 cancels a bus access wait state corresponding to a cancel signal 207, cancels a **write back** request to the bus interface 203 of the **copy back** type cache memory 202 corresponding to a cancel signal 208 and cancels the external access of the instruction execution part 201 corresponding to the cancel signal 208.

14/5/17 (Item 17 from file: 347)
DIALOG(R)File 347:JAPIO

03329166 **Image available**
CONSISTENCY HOLDING SYSTEM FOR **WRITE - THROUGH** CACHE

PUB. NO.: 02-304666 [JP 2304666 A]
PUBLISHED: December 18, 1990 (19901218)
INVENTOR(s): OZAKI KEIJI
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 01-126324 [JP 89126324]
FILED: May 19, 1989 (19890519)
INTL CLASS: [5] **G06F-015/16** ; **G06F-012/08**
JAPIO CLASS: 45.4 (INFORMATION PROCESSING -- Computer Applications); 45.2 (INFORMATION PROCESSING -- Memory Units)
JOURNAL: Section: P, Section No. 1174, Vol. 15, No. 88, Pg. 116, March 04, 1991 (19910304)

ABSTRACT

PURPOSE: To efficiently execute holding of the consistency between each **write - through** cache by providing a consistency holding part having a bus monitoring **part** , a **data** fetch **part** and a **cache data** changing **part** against each **write - through cache** .

CONSTITUTION: A bus monitoring part 131 or 231 in a consistency holding part 13 or 23 in a central processing unit 1 or 2 executes such a processing as shown hereinbelow. By always monitoring a memory bus 4, whether a write instruction to a main storage device 3 is issued or not is decided. As a result of this decision, in the case the write instruction to the device 3 is issued, whether a write destination address related to its write instruction is contained in a memory block held by a **write - through** cache 12 or not is decided. As a result of this decision, in the case the write destination address related to its write instruction is contained in the cache 12, it is requested to fetch write data (data related to the write instruction concerned) from the memory bus 4, to a data fetch part 132 (write data fetch request is executed).

```
Set     Items   Description
S1     287368   CACH? OR BUFFER? OR TEMPORAR?(N)(STOR? OR SAVE?)
S2     125680   (SPECIFIC? OR PARTIAL OR PART OR CERTAIN? OR DESIGNAT? OR -
             SEGMENT?)(2N)(CODE OR PROGRAM? OR INSTRUCTION? OR DATA OR SUB-
             PROGRAM?)
S3     726810   MARKER? OR FLAG OR FLAGS OR TAG OR TAGS OR INDICAT? OR LAB-
             EL?
S4       6324   (APPEND? OR ADD()ON OR INSERT?)(2N)(BIT OR BITS OR S3)
S5        996   (COPY OR WRITE)()(BACK OR THROUGH) OR WRITETHROUGH OR WRIT-
             EBACK OR COPYBACK? OR COPYTHROUGH?
S6       9091   COMPILE?
S7          0   S1 AND S2 AND S3 AND S4 AND S5
S8         19   S1 AND S2 AND S4
S9          2   S1 AND S5 AND S6
S10        14   S1(4N)S2 AND S5
S11        35   S8 OR S9 OR S10
S12        20   S11 AND IC=G06F?
S13        20   IDPAT (sorted in duplicate/non-duplicate order)
S14        20   IDPAT (primary/non-duplicate records only)
S15    923945   LEVEL? OR MULTILEVEL? OR TIER? OR HIERARCH?
S16       516   S1 AND S2 AND S15
S17         7   S16 AND S6
S18         0   S1 AND S3 AND S4 AND S5
S19         0   S1 AND S4 AND S5
S20       298   S1 AND S4
S21        35   S20 AND (S6 OR S15)
S22         3   S21 AND IC=G06F?
S23         5   S17 AND IC=G06F?
S24         8   S22 OR S23
S25         8   IDPAT (sorted in duplicate/non-duplicate order)
S26         8   IDPAT (primary/non-duplicate records only)
File 347:JAPIO Nov 1976-2004/May(Updated 040903)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &UP=200457
         (c) 2004  Thomson Derwent
?
```

26/5/6 (Item 6 from file: 347)
DIALOG(R)File 347:JAPIO

03874223 **Image available**
INSTRUCTION **CACHE** SYSTEM FOR **HIERARCHICAL** INSTRUCTION CONTROL

PUB. NO.: 04-239323 [JP 4239323 A]
PUBLISHED: August 27, 1992 (19920827)
INVENTOR(s): YOKOYAMA YASUSHI
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 03-002430 [JP 912430]
FILED: January 14, 1991 (19910114)
INTL CLASS: [5] **G06F-009/38** ; **G06F-009/26** ; **G06F-009/28** ; **G06F-009/32**

JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)
JOURNAL: Section: P, Section No. 1465, Vol. 17, No. 7, Pg. 82, January 07, 1993 (19930107)

ABSTRACT

PURPOSE: To reduce the overhead loss by using the instructions of 1st and 2nd groups to **compile** a software program.

CONSTITUTION: An **instruction** prefetch control **part** 6 gives a request to a 1st system **instruction** **cache** control **part** 311 to take an instruction out of a software program 12 through an address included in a main storage storing a 1st group instruction following a 2nd group **instruction** . Thus the **part** 6 retrieves a hit state. When a hit state is confirmed, the data are immediately stored in an instruction register 5 from a 1st system instruction **cache** 310. Then the execution of an instruction program is started. In a mishit state, however, the data are read out of the storage 1 via a main storage access control part 2 under the control of the part 311. Then the date are loaded in a block of the **cache** 310 and also in an instruction register 5, and the instruction of the instruction program is carried out. In such a constitution, the overhead loss can be reduced.

```
Set     Items   Description
S1     333985   CACH? OR BUFFER? OR TEMPORAR?(N)(STOR? OR SAVE?)
S2     132373   (SPECIFIC? OR PARTIAL OR PART OR CERTAIN? OR DESIGNAT? OR -
             SEGMENT?)(2N)(CODE OR PROGRAM? OR INSTRUCTION? OR DATA OR SUB-
             PROGRAM?)
S3    4282379   MARKER? OR FLAG OR FLAGS OR TAG OR TAGS OR INDICAT? OR LAB-
             EL?
S4       3848   (APPEND? OR ADD()ON OR INSERT?)(2N)(BIT OR BITS OR S3)
S5        930   (COPY OR WRITE)()(BACK OR THROUGH) OR WRITETHROUGH OR WRIT-
             EBACK OR COPYBACK? OR COPYTHROUGH?
S6    4469300   LEVEL? OR HIERARCH? OR TIER? OR MULTILEVEL?
S7     180535   COMPIL?
S8          1   S1 AND S2 AND S4
S9          1   S1 AND S4 AND S5
S10        10   S1 AND S4 AND S6
S11       127   S1 AND S2 AND S3
S12        10   S11 AND S7
S13         0   S11 AND S5
S14       101   S1 AND S5 AND S3
S15        49   S14 AND (S6 OR S7)
S16        70   S8 OR S9 OR S10 OR S12 OR S15
S17        43   RD (unique items)
S18        35   S17 NOT PY>2000
File   8:Ei Compendex(R) 1970-2004/Aug W5
         (c) 2004 Elsevier Eng.  Info. Inc.
File  35:Dissertation Abs Online 1861-2004/Aug
         (c) 2004 ProQuest Info&Learning
File 202:Info. Sci. & Tech. Abs. 1966-2004/Jul 12
         (c) 2004 EBSCO Publishing
File  65:Inside Conferences 1993-2004/Sep W1
         (c) 2004 BLDSC all rts. reserv.
File   2:INSPEC 1969-2004/Aug W5
         (c) 2004 Institution of Electrical Engineers
File  94:JICST-EPlus 1985-2004/Aug W2
         (c)2004 Japan Science and Tech Corp(JST)
File 111:TGG Natl.Newspaper Index(SM) 1979-2004/Sep 08
         (c) 2004 The Gale Group
File 233:Internet & Personal Comp. Abs. 1981-2003/Sep
         (c) 2003 EBSCO Pub.
File   6:NTIS 1964-2004/Aug W4
         (c) 2004 NTIS, Intl Cpyrght All Rights Res
File 144:Pascal 1973-2004/Aug W5
         (c) 2004 INIST/CNRS
File  34:SciSearch(R) Cited Ref Sci 1990-2004/Aug W5
         (c) 2004 Inst for Sci Info
File  99:Wilson Appl. Sci & Tech Abs 1983-2004/Jul
         (c) 2004 The HW Wilson Co.
File  95:TEME-Technology & Management 1989-2004/Jun W1
         (c) 2004 FIZ TECHNIK
```

18/5/2 (Item 2 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

04850633 E.I. No: EIP97103881117
Title: Decoupled modified-bit cache
Author: Takahashi, Masafumi; Oba, Nobuyuki; Kobayashi, Hiroaki; Nakamura, Tadao
Corporate Source: Tohoku Univ, Jpn
Source: Systems and Computers in Japan v 28 n 6 Jun 15 1997. p 49-59
Publication Year: 1997
CODEN: SCJAEP ISSN: 0882-1666
Language: English
Document Type: JA; (Journal Article) Treatment: T; (Theoretical)
Journal Announcement: 9712W2
Abstract: **Cache** memory not only allows one to decrease average memory access time, but also relieves bus traffic and decreases bus latency. In this paper, to further relieve bus traffic, a **write - back cache** memory (DMC, Decoupled Modified-bit **Cache**) is proposed that provides data modification at byte **level** . DMC supports selective **write - back** of only modified data to memory which contributes to further relief of bus traffic. To avoid considerable requirements for additional hardware in implementing DMC, a method is proposed to separate status bits that **indicate** data modification from the **cache** , allocating them as necessary. Benchmark tests with a variety of applications were performed to validate DMC. The results show that, with an additional 3% of the **cache** memory allocated as memory cells for status bits, memory usage intensity and data flow through the bus are reduced by about 35% and 10%, respectively, when compared to a conventional **write - back cache** . (Author abstract) 11 Refs.
Descriptors: **Buffer** storage; Storage allocation (computer); Data transfer
Identifiers: Decoupled modified bit **cache** (DMC); **Write back cache**
Classification Codes:
722.1 (Data Storage, Equipment & Techniques); 723.2 (Data Processing)
722 (Computer Hardware); 723 (Computer Software)
72 (COMPUTERS & DATA PROCESSING)

18/5/3 (Item 3 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03671352 E.I. No: EIP93050792255
Title: Second- level cache **for the 80486 using a PLD-based** cache **controller**
Author: Legenhausen, Jay; Sevalia, Piyush
Corporate Source: Cypress Semiconductor, San Jose, CA, USA
Conference Title: Wescon '92
Conference Location: Anaheim, CA, USA Conference Date: 19921117-19921119
Sponsor: IEEE; ERA
E.I. Conference No.: 18412
Source: Wescon Conference Record v 36 1992. Publ by Wescon, Los Angeles, CA, USA. p 76-83
Publication Year: 1992
CODEN: WCREDI
Language: English
Document Type: CA; (Conference Article) Treatment: T; (Theoretical); A; (Applications)
Journal Announcement: 9309W2
Abstract: This paper describes a complete, high-performance, second-**level** **cache** system for a 33-MHz 80486 that is optimized for both high performance and low component in a single PLD that controls a 128-Kbyte, direct-mapped, **write - through** **cache** that supplements the 486's on-chip 8-Kbyte **cache** . The overall system can be divided into four main sections: the CPU, the **cache** , the **cache** **tags** , and the **cache** controller. The CPU is a 33-MHz Intel 80486; the 128-Kbyte **cache** consists of four 44-pin, synchronous, 32K multiplied by 9 **cache** RAMs; the **cache** **tag** consists of two CY7B181 4K multiplied by 18 **cache** **tags** ; the **cache** controller is a single CY7C344 PLD. The entire system was simulated and correct operation verified using Viewlogic's simulation tools. The first part of this document discusses the four main system blocks, starting with the memory interface to the 486. Following that, the **cache** controller implementation is described. The last section of the paper describes the simulation method used to verify correct system operation. (Author abstract) 1 Ref.
Descriptors: *Data storage equipment; Random access storage; Microcomputers; Minicomputers; Computer circuits; Microprocessor chips; Digital integrated circuits; Simulation
Identifiers: **Cache** memory; **Cache** **tag** ; PLD based controller; Memory interface
Classification Codes:
722.1 (Data Storage, Equipment & Techniques); 721.2 (Logic Elements); 721.3 (Computer Circuits); 713.5 (Other Electronic Circuits)
722 (Computer Hardware); 721 (Computer Circuits & Logic Elements); 713 (Electronic Circuits)
72 (COMPUTERS & DATA PROCESSING); 71 (ELECTRONICS & COMMUNICATIONS)

18/5/7 (Item 7 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03038779 E.I. Monthly No: EIM9103-012695
Title: The performance impact of block sizes and fetch strategies.
Author: Przybylski, Steven
Corporate Source: MIPS Computer Systems, Sunnyvale, CA, USA
Conference Title: Proceedings of the 17th Annual International Symposium on Computer Architecture
Conference Location: Seattle, WA, USA Conference Date: 19900528
Sponsor: IEEE Computer Soc; ACM SIGARCH
E.I. Conference No.: 14053
Source: Conference Proceedings - Annual Symposium on Computer Architecture. Publ by IEEE, Computer Society, Los Alamitos, CA, USA (IEEE cat n 90CH2887-8). p 160-169
Publication Year: 1990
CODEN: CPAADU ISSN: 0149-7111 ISBN: 0-8186-2047-1
Language: English
Document Type: PA; (Conference Paper) Treatment: X; (Experimental)
Journal Announcement: 9103
Abstract: The interactions between a **cache** 's block size, fetch size, and fetch policy from the perspective of maximizing system- **level** performance are explored. It has been previously noted that, given a simple fetch strategy, the performance optimal block size is almost always four or eight words. If there is even a small cycle time penalty associated with either longer blocks or fetches, then the performance optimal size is noticeably reduced. In split **cache** organizations, where the fetch and block sizes of instruction and data **caches** are all independent design variables, instruction **cache** block size and fetch size should be the same. For the workload and **write - back** write policy used in this trace-driven simulation study, the instruction **cache** block size should be about a factor of 2 greater than the data **cache** fetch size, which in turn should be equal to or double the data **cache** block size. The simplest fetch strategy of fetching only on a miss and stalling the CPU until the fetch is complete works well. Complicated fetch strategies do not produce the performance improvements **indicated** by the accompanying reductions in miss ratios because of limited memory resources and a strong temporal clustering of **cache** misses. For the environments simulated, the most effective fetch strategy improved performance by between 1.7% and 4.5% over the simplest strategy described above. 21 Refs.
Descriptors: *COMPUTER SYSTEMS, DIGITAL--*Multiprocessing; COMPUTER ARCHITECTURE; COMPUTER SIMULATION; DATA STORAGE UNITS
Identifiers: **CACHE** MEMORIES; MEMORY MANAGEMENT; TRACE DRIVEN SIMULATION
Classification Codes:
722 (Computer Hardware); 723 (Computer Software)
72 (COMPUTERS & DATA PROCESSING)

18/5/8 (Item 8 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

01989932 E.I. Monthly No: EI8607056100 E.I. Yearly No: EI86021011
Title: RECOMPUTING CYCLIC REDUNDANCY CODE CHECK BITS.
Author: Anon
Source: IBM Technical Disclosure Bulletin v 28 n 11 Apr 1986 p 4768-4770
Publication Year: 1986
CODEN: IBMTAA ISSN: 0018-8689
Language: ENGLISH
Document Type: JA; (Journal Article) Treatment: A; (Applications)
Journal Announcement: 8607
Abstract: In some computer subsystem architectures, all or **part** of the **data** from a direct-access storage device is **temporarily stored** before being transmitted to the system. To assure data integrity, Cyclic Redundancy Code (CRC) check **bits** are typically **appended** at the end of the data field by the transmitting end, and stored at the receiving end. If errors are detected, an attempt is made to correct them before sending them to the channel. However, once any part of the field is changed because of correction or updating, the CRC bits are no longer valid, and have to be updated to reflect the changed data. In accordance with the new method described, CRC check bits for data which have been updated are themselves updated without having to clock the data through a CRC generator. As the length of the data field becomes relatively large, improvement in execution time becomes more significant.
Descriptors: *COMPUTER ARCHITECTURE--*Performance; CODES, SYMBOLIC
Identifiers: CYCLIC REDUNDANCE; CODE CHECK BITS; SUBSYSTEM ARCHITECTURES; DATA INTEGRITY; EXECUTION TIME
Classification Codes:
722 (Computer Hardware); 723 (Computer Software)
72 (COMPUTERS & DATA PROCESSING)

18/5/9 (Item 9 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

01097828 E.I. Monthly No: EI8203018859 E.I. Yearly No: EI82017560
Title: ANALYSIS OF MULTIPROCESSOR CACHE **ORGANIZATIONS WITH ALTERNATIVE MAIN MEMORY UPDATE POLICIES.**
Author: Yen, W. C.; Fu, K. S.
Corporate Source: Purdue Univ, West Lafayette, Indiana, USA
Source: Conf Proc Annu Symp Comput Archit 8th, Minneapolis, Minn, USA, May 12-14 1981. Publ by IEEE Comput Soc Press (Cat n 81CH1593-3), Los Alamitos, Calif, USA p 89-105
Publication Year: 1981
CODEN: CPAADU ISSN: 0149-7111
Language: ENGLISH
Journal Announcement: 8203
Abstract: **Cache** memory has played a significant role in the memory **hierarchy** and has been used extensively in large systems and minisystems. The effectiveness of **cache** memories with alternative main memory update policies in a multiprocessor system is a major concern of this study. The performances of **write - through** with write-allocation or no-write allocation, **buffered write - through , flag** -swap, and **buffered flag** -swap policies have been analyzed. Because of the dominating cost of the interface between processors and main memory modules in the multiprocessor system, the effect of varying the bus width or block size has also been considered. Queuing models were developed to analyze these alternative organizations, and results predicted by the models were validated by a set of simulations. 30 refs.
Descriptors: *COMPUTER ARCHITECTURE; COMPUTER SYSTEMS, DIGITAL--Multiprocessing
Identifiers: **CACHE** MEMORY
Classification Codes:
723 (Computer Software); 722 (Computer Hardware)
72 (COMPUTERS & DATA PROCESSING)

01483454 ORDER NO: AADAA-I9615194

CACHE MEMORY DESIGN AND PERFORMANCE ISSUES IN SHARED-MEMORY MULTIPROCESSORS

Author: MOUNES-TOUSSI, FARNAZ
Degree: PH.D.
Year: 1995
Corporate Source/Institution: UNIVERSITY OF MINNESOTA (0130)
Major Adviser: DAVID J. LILJA
Source: VOLUME 57/01-B OF DISSERTATION ABSTRACTS INTERNATIONAL.
PAGE 584. 158 PAGES
Descriptors: ENGINEERING, ELECTRONICS AND ELECTRICAL ; COMPUTER SCIENCE
Descriptor Codes: 0544; 0984

The widening gap between the speed of processors and main memory has encourage the use of **cache** memories to reduce the average memory access time by exploiting the spatial and temporal locality of memory references in a program. In shared-memory multiprocessors, however, data sharing reduces the effectiveness of **cache** memories by introducing the **cache** coherence problem and the inherent coherence overhead. The many solutions to the **cache** coherence problem attempt to reduce the coherence overhead by using different mechanism to enforce coherence, to detect accesses to stale data, and to reduce the impact of false-sharing. In this thesis, we examine the performance effect of these different mechanisms. We discuss how the coherence overhead depends on the sharing characteristics of a program and define a model for characterizing the sharing behavior of parallel programs. We also propose and evaluate a partially-valid **write - through** write-merged coherence mechanism to tolerate false-sharing. This mechanism by itself does not entirely eliminate redundant **write - through** , however, so we also define a **compiler** algorithms to eliminate some of the redundant **write - through** to memory.

Using simulations and our model, we identify the high processor-memory network traffic as the main problem associated with an updating coherence enforcement strategy, and the high number of misses as the main problem associated with an invalidating coherence enforcement strategy. We also show that the severe performance penalty resulting from inaccurate **compile** -time analysis in a software-only coherence detection mechanism favors the use of a hardware-only or a combined hardware-software mechanism. Our study of false-sharing **indicates** that the performance penalty of false-sharing is severe when a program has a large amount of sharing and a fine granularity of sharing. Our studies of a partially-valid **write - through** **cache** with and without a merging write- **buffer** emphasize the difficulty of using a hardware mechanism to adjust to the programs' sharing characteristics. Consequently, we address this difficulty using **compile** -time analysis in conjunction with a hardware mechanism to enhance performance.

01311312 ORDER NO: AAD93-26431

SPECIALIZED CACHES **TO IMPROVE DATA ACCESS PERFORMANCE (DATA MEMORY** HIERARCHY , **WRITE** CACHES , **SUBWORD** CACHES , **FETCH** CACHES , **MEMORY)**

Author: BRAY, BRIAN KENWORTHY

Degree: PH.D.
Year: 1993
Corporate Source/Institution: STANFORD UNIVERSITY (0212)
Adviser: MICHAEL J. FLYNN
Source: VOLUME 54/05-B OF DISSERTATION ABSTRACTS INTERNATIONAL.
PAGE 2647. 156 PAGES
Descriptors: ENGINEERING, ELECTRONICS AND ELECTRICAL
Descriptor Codes: 0544

High performance processor organizations place large demands on the data memory **hierarchy** . The data bandwidth requirements of a processor can be a serious performance constraint. As a result, **cache** memory is a very important element in the implementation of a high performance computer. **Caching** a subset of main memory in faster memory provides the processor with high bandwidth and low latency data. The utility of **caching** is determined by the **cache** size, organization, and speed.

The focus of this dissertation is to improve the performance of a data **cache** by the addition of small specialized **caches** . By analyzing the spatial and temporal locality in the data reference stream at various points in the data memory **hierarchy** , we have developed a mixture of specialized **caches** to improve the data memory **hierarchy** .

We have developed and analyzed write **caches** , **tag caches** , subword **caches** , fetch **caches** , and a two- **level** windowed register file. Each specialized **cache** reduces the bandwidth and/or latency demands on the data memory **hierarchy** . Write **caches** reduce **write - through cache** traffic. **Tag caches** reduce **tag** checks to **write - back caches** . Subword **caches** enhance the benefit of removing the subword hardware from the primary **cache** access path. Fetch **caches** reduce read latency or increase read bandwidth. A two- **level** windowed register file provides large capacity without impacting cycle time. Thus, a mixture of specialized **caches** has been developed to improve the data memory **hierarchy** .

18/5/12 (Item 3 from file: 35)
DIALOG(R)File 35:Dissertation Abs Online

779523 ORDER NO: AAD82-12495
CACHE **STRUCTURES BASED ON THE EXECUTION STACK FOR HIGH** LEVEL **LANGUAGES**
Author: BORGWARDT, PETER ARTHUR
Degree: PH.D.
Year: 1981
Corporate Source/Institution: UNIVERSITY OF WASHINGTON (0250)
Source: VOLUME 42/12-B OF DISSERTATION ABSTRACTS INTERNATIONAL.
PAGE 4859. 231 PAGES
Descriptors: COMPUTER SCIENCE
Descriptor Codes: 0984

The purpose of this work is to search for hardware organizations that might speed up memory referencing for block-structured languages such as Pascal. Our hypothesis is that building basic knowledge about execution stacks into the hardware will improve this referencing. The problem divides into two parts: efficient calculation of memory addresses and efficient movement of data between memory and CPU. A stack address generator attacks the first problem. Various schemes for **caching** data attempt to address the second: varied **cache** -write policies (CPU writing **cache** in the absence of a **tag** match), a program prefetch **buffer** , a top of stack minicache that limits writethroughs of temporaries, and a two-access **cache** than can both be fetched from and stored to in a single microcycle. Several designs are proposed, microcoded to emulate Pascal P-code, and their performances compared on twenty-two Pascal test programs.

The addition of the stack address generator results in a 40 to 55% speedup. Varying the **cache** -write policy has little effect. Program prefetching shortens the microcode so much that CPU waits for memory contention and bus protocols are a significant problem. A **writeback** policy would help this; the minicache which creates a **writeback** policy only for top of stack temporaries reduces much of the contention. Stack registers were not very beneficial when prefetching was not used. The minicache, program prefetching, and the two-access **cache** together give an additional 40 to 50% speedup. However, in the microcode the top of stack is either the source or destination of all two-accesses. Hence the two-access performance can be attained by merely providing the top of stack minicache the ability to communicate with the main **cache** within a single microcyle.

18/5/13 (Item 1 from file: 2)
DIALOG(R)File 2:INSPEC

6404291 INSPEC Abstract Number: C1999-12-5320G-011
Title: Selective-set-invalidation (SSI) for soft-error-resilient cache **architecture**
Author(s): Hwang, S.H.; Choi, G.S.
Author Affiliation: Dept. of Electr. Eng., Texas A&M Univ., College Station, TX, USA
Journal: Computer Architecture News vol.27, no.3 p.4-9
Publisher: ACM,
Publication Date: June 1999 Country of Publication: USA
CODEN: CANED2 ISSN: 0163-5964
SICI: 0163-5964(199906)27:3L.4:SISE;1-M
Material Identity Number: B580-1999-004
Language: English Document Type: Journal Paper (JP)
Treatment: Practical (P)
Abstract: The paper proposes a novel **cache** -memory design for soft-error silence, and verifies the design through a simulation that uses realistic system and software model (J.L. Hennessy and D.A. Patterson, 1996). The SSI design is a combination of an n-bit error detector and a fast circuit that allows real-time-forced invalidation of corrupted data sets. The current design supports the **write - through caching** policy and will be extended for the **write - back** policy. To verify the effectiveness of the proposed design approach, mixed-mode simulations are conducted that **insert** soft-errors (**bit** -flips) into the **cache** -memory model. The simulation commences while running different class of programs (ALU-intensive and branch-intensive programs) designed to stress several key functions of the target system. System- **level** failure modes triggered by the soft errors are observed and all inserted errors are recovered by SSI scheme. The performance and layout-area overheads are also quantified. The worst-case performance/time overhead of the SSI scheme is approximately 9% while the lay-out-area overhead is less than 0.5%. (19 Refs)
Subfile: C
Descriptors: **cache** storage; fault tolerant computing; memory architecture; real-time systems; virtual machines
Identifiers: selective-set-invalidation; soft-error-resilient **cache** architecture; novel **cache** -memory design; soft-error silence; realistic system; software model; SSI design; n-bit error detector; fast circuit; real-time-forced invalidation; corrupted data sets; **write - through caching** policy; **write - back** policy; design approach; mixed-mode simulations; **cache** -memory model; branch-intensive programs; key functions ; system- **level** failure modes; soft errors; inserted errors; layout-area overheads; worst-case performance/time overhead
Class Codes: C5320G (Semiconductor storage); C6120 (File organisation); C5310 (Storage system design); C7430 (Computer engineering); C5470 (Performance evaluation and testing)

write-back cache

Last modified: Friday, November 09, 2001

A caching method in which modifications to data in the cache aren't copied to the cache source until absolutely necessary. Write-back caching is available on many microprocessors, including all Intel processors since the 80486. With these microprocessors, data modifications (e.g., write operations) to data stored in the L1 cache aren't copied to main memory until absolutely necessary. In contrast, a *write-through cache* performs all write operations in parallel -- data is written to main memory and the L1 cache simultaneously.

Write-back caching yields somewhat better performance than write-through caching because it reduces the number of write operations to main memory. With this performance improvement comes a slight risk that data may be lost if the system crashes.

A write-back cache is also called a *copy-back cache.*